AF4
JFW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
Before the Board of Patent Appeals and Interferences

In re Patent Application of

FAIRMAN et al

Serial No. 09/555,929

Filed: June 6, 2000
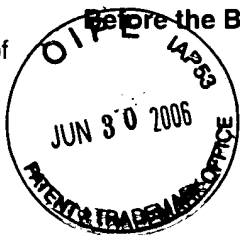
Title: DATA COMMUNICATIONS

Atty Dkt. 36-1334
                C#   M#

TC/A.U.: 2135

Examiner: Ha, L.

Date: June 30, 2006

[Stamp: JUN 30 2006]

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

☐ **Correspondence Address Indication Form Attached.**

☐ **NOTICE OF APPEAL**
   Applicant hereby **appeals** to the Board of Patent Appeals and Interferences
   from the last decision of the Examiner twice/finally rejecting $500.00 (1401)/$250.00 (2401) $
   applicant's claim(s).

☒ An appeal **BRIEF** is attached in the pending appeal of the
   above-identified application $500.00 (1402)/$250.00 (2402) $ 500.00

☐ Credit for fees paid in prior appeal without decision on merits   -$ (   )

☐ A reply brief is attached.   **(no fee)**

☐ Petition is hereby made to extend the current due date so as to cover the filing date of this
   paper and attachment(s)   One Month Extension $120.00 (1251)/$60.00 (2251)
                                        Two Month Extensions $450.00 (1252)/$225.00 (2252)
                                        Three Month Extensions $1020.00 (1253/$510.00 (2253)
                                        Four Month Extensions $1590.00 (1254/$795.00 (2254) $

   ☐ "Small entity" statement attached.

   Less   month extension previously paid on   -$(   )

                                                                        TOTAL FEE ENCLOSED $ 500.00

Any future submission requiring an extension of time is hereby stated to include a petition for such time extension.
The Commissioner is hereby authorized to charge any <u>deficiency</u>, or credit any overpayment, in the fee(s) filed, or
asserted to be filed, or which should have been filed herewith (or with any paper hereafter filed in this application by this
firm) to our **Account No. 14-1140**. A <u>duplicate</u> copy of this sheet is attached.

901 North Glebe Road, 11th Floor
Arlington, Virginia 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100
RYM:sl

NIXON & VANDERHYE P.C.
By Atty: Raymond Y. Mah, Reg. No. 41,426

Signature: _____

In re Patent Application of

FAIRMAN et al.   Atty. Ref.: 36-1334

Serial No. 09/555,929   Group: 2135

Filed: June 6, 2000   Examiner: Ha, L.

For: DATA COMMUNICATIONS

*   *   *   *   *   *   *   *   *   *   *

June 30, 2006

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

Appellant hereby **appeals** to the Board of Patent Appeals and Interferences from

the last decision of the Examiner.

TABLE OF CONTENTS

(I)   <u>REAL PARTY IN INTEREST</u>

The real party in interest is British Telecommunications public limited company, a corporation of the country of England.

(II)   <u>RELATED APPEALS AND INTERFERENCES</u>

The appellant, the undersigned, and the assignee are not aware of any related appeals, interferences, or judicial proceedings (past or present), which will directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

(III) STATUS OF CLAIMS

Claims 1, 4-26 and 29-33 are pending and have been rejected under 35 U.S.C.

§102 as allegedly being anticipated by Iverson et al (U.S. Patent No. 5,852,664,

hereinafter "Iverson"). No claims have been substantively allowed.

(IV) <u>STATUS OF AMENDMENTS</u>

An Amendment/Response was filed on March 27, 2006 (i.e., after the date of the

Final Rejection). The Advisory Action mailed April 12, 2006 indicated that the

amendments of the March 27, 2006 Amendment/Response would not be entered for

purposes of appeal.

(V) <u>**SUMMARY OF CLAIMED SUBJECT MATTER**</u>

The invention of the claims relates to a service provision system/method for use in

distributed processing environments. A listing of each independent claim, each

dependent claim argued separately and each claim having means plus function language

is provided below including exemplary reference(s) to page and line number(s) of the

specification.

1.   A method of distributing digitally encoded data, comprising:

a)   dividing said data into a multiplicity of frames, [pg. 6, ll. 28-31]

b)   encrypting said frames, [pg. 6, ll. 31-33]

c)   distributing multiple copies of the said data frames to a multiplicity of

users, each frame being distributed with a control field, [pg. 6, ll. 18-22; pg. 7, ll. 1-2 and

17-25; pg. 14, l. 21 - pg. 15, l. 15]

d)   communicating a seed value for key generation to respective secure

modules located at each of the multiplicity of users, [pg. 7, ll. 8-11; pg. 10, ll. 7-9]

e)   decrypting the data frames at respective users using keys derived from the

seed value communicated to the secure module, the secure module being arranged to

enable decryption of a respective frame only when said control field has been passed to

the secure module, [pg. 7, ll. 6-13; pg. 8, 27-33]

f)   passing a control message, for modifying and controlling the availability of

keys, in the control field to the secure module at a selected one or more users, and [pg. 7,

ll. 17-25]

g)   at the secure module of the or each selected user, in response to the said

control message, controlling the availability of keys generated from the said seed value,

thereby controlling access by the users to the said data. [pg. 7, ll. 1-12]

4.   A method according to claim 1, in which each data frame includes a frame

identity field, and each key generated by the secure module is specific to one frame

identified by the said field. [pg. 3, ll. 19-23; pg. 18, ll. 13-21]

5.   A method according to claim 1, in which the step of distributing multiple

copies of the said data comprises multicasting packets of data via a communications

network to the plurality of users. [pg. 1, ll. 4-5; pg. 2, ll. 25-26; pg. 6, ll. 20-25]

6.   A method according to claim 1, in which the control message is distributed

with a data frame to the multiplicity of users, a user identity field identifying a selected

user or group of users is included in the control message, and the control message is acted

on only by the user or group of users identified by the said user identity field. [pg. 4, ll.

17-20; pg. 15, l. 24 - pg. 16, l. 2]

7.   A method according to claim 1, in which the control message includes a

stop flag, and in response to the stop flag the generation of keys at the or each selected

user is stopped. [pg. 14, l. 21 - pg. 15, l. 4]

8.   A method according to claim 1, including returning a response signal from

the secure module to the source of the control message. [pg. 4, ll. 21-24; pg. 14, l. 21 -

pg. 15, l. 6]

9.   A method according to claim 8, in which the control message includes a

contact sender flag, and the step of returning a response signal from the secure module is

carried out when the contact sender flag is set. [pg. 4, ll. 21-24; pg. 14, l. 21 - pg. 15, l. 6]

10.   A method according to claim 8, including transmitting a further control

message to the user on receipt of the said response signal. [pg. 4, ll. 21-24; pg. 14, l. 21 -

pg. 15, l. 6]

11.   A method of operating a customer terminal in a data communications

system, the method comprising:

a)   receiving at the customer terminal a multiplicity of encrypted data frames,

each with a control field; [pg. 6, ll. 18-22; pg. 7, ll. 1-2 and 17-25; pg. 14, l. 21 - pg. 15, l.

15]

b)   receiving at the customer terminal a seed value for key generation; [pg. 7,

ll. 8-11; pg. 10, ll. 7-9]

c)   passing the said seed value for key generation to a secure module

located at the customer terminal; [pg. 7, ll. 8-11]

d)   generating in the secure module using the seed value keys for the

decryption of data frames; [pg. 7, ll. 8-11]

e)   decrypting using the said keys only those respective data frames for which

a control field has been received; [pg. 7, ll. 6-13; pg. 8, 27-33]

f)   passing to the said secure module a control message received in the control

field; and [pg. 7, ll. 17-25]

g)   in response to the said control message, controlling the availability of keys

generated using the said seed value and thereby controlling access by the user of the

customer terminal to data received at the customer terminal. [pg. 7, ll. 1-12]

12.   A data communications system comprising:

a)   a remote data source arranged to output a plurality of frames; [pg. 6, ll. 18-

20]

b)   encryption means for encrypting the plurality of frames with different

respective keys; [pg. 6, ll. 31-34]

c)   a communications channel arranged to distribute multiple copies of the

encrypted data frames, each with a control field; [pg. 6, ll. 18-22; pg. 7, ll. 1-2 and 17-

25; pg. 14, l. 21 - pg. 15, l. 15]

d)   a multiplicity of customer terminals arranged to receive from the

communications channel respective copies of the encrypted data frames with the control

fields; [pg. 6, ll. 18-34]

e)   a key generator located at a customer terminal and programmed to generate

from a seed value keys for use in decrypting data frames; [pg. 7, ll. 8-11; pg. 10, ll. 7-9]

f)   key control means connected to the key generator, the key control means

comprising:

an interface for receiving the control fields ; and

control means arranged to only release keys for decrypting those respective

frames for which a control field is received and being arranged to, in response to

the said control messages in the control fields, control the availability to the user

of keys generated from the seed value; and [pg. 7, ll. 17-25]

g)   decryption means connected to the key generator and arrange to decrypt the

data frames received at the customer terminal from the communications channel. [pg. 7,

ll. 6-13; pg. 8, 27-33]

13.   A data communications system according to claim 12, in which the

communications channel is a packet-switched data network. [pg. 1, ll. 4-5; pg. 2, ll. 25-

26; pg. 6, ll. 20-25]

14.   A customer terminal for use in a method according to claim 1, the customer

terminal comprising:

a)   a data interface for connection to a data communications channel; [pg. 8, ll.

25-28]

b)   a key generator programmed to generate from a seed value keys for use in decrypting data frames: [pg. 7, ll. 8-11; pg. 10, ll. 7-9]

c)   decryption means connected to the data interface and the key generator and arranged to decrypt data frames received via the data interface; and [pg. 7, ll. 6-13; pg. 8, ll. 27-33]

d)   key control means connected to the key generator, the key control means comprising:

an interface for receiving control fields; and

control means arrange to only release keys for decrypting those respective data frames received with a control field;

the control means being arranged to in response to control messages in the control fields, control the availability to the user keys generated from the seed value. [pg. 7, ll. 17-25]

15.   A data server for use in method according to claim 1, the data server comprising:

a)   a data interface for connection to a data communications channel; [Fig. 1, pg. 6, ll. 18-20]

b)   means for outputting encrypted data frames with control fields via the data interface onto the communications channel for receipt by a multiplicity of customer terminals; [pg. 6, ll. 31-33]

c)   means for outputting the control fields having control messages onto a data

communications channel for controlling the operation of key generators at customer

terminals. [pg. 6, ll. 18-22; pg. 7, ll. 1-2 and 17-25; pg. 14, l. 21 - pg. 15, l. 15]

16.   A method according to claim 1, including generating keys from the seed

value by iterated operations on the seed value by selected ones of a plurality of

predetermined functions. [pg. 7, ll. 8-11; pg. 10, ll. 7-9; pg. 14, ll. 14-15]

21.   A method according to claim 1, including applying different characteristic

variations to data decrypted at different respective customer terminals. [pg. 4, ll. 24-29]

22.   A method according to claim 1, including a plurality of remote data

sources, each outputting a respective plurality of frames. [pg. 17, ll. 1-6]

23.   A method according to claim 22, in which the customer terminal receives a

primary seed value common to different respective data streams from the plurality of data

sources, and derives from the common primary key a plurality of different respective

secondary seed values for decrypting frames from different respective data sources. [pg.

17, ll. 10-15]

24.   A method according to claim 23, in which data received from different data

sources includes different respective source identity values, and the respective secondary

seed value is generated from the primary seed value by modifying the primary seed value

with the source identity value. [pg. 17, ll. 20-23]

25.   A method according to claim 1, in which each data frame includes a frame

type field. [pg. 8, ll. 6-16]

26.   A method according to claim 25, including storing a receipt including data

from the frame type field. [pg. 8, ll. 6-14]

29.   A method as in claim 1, wherein the control message received by the secure

module causes the secure module to cease releasing keys. [pg. 14, l. 21 - pg. 15, l. 4]

30.   A method as in claim 11, wherein the control message received by the

secure module causes the secure module to cease releasing keys. [pg. 14, l. 21 - pg. 15, l.

4]

31.   A data communications system as in claim 12, wherein the control message

received by the key control means causes the key control means to cease releasing keys.

[pg. 14, l. 21 - pg. 15, l. 4]

32.   A method as in claim 1, wherein each of the frames is encrypted with a

different key. [pg. 6, ll. 33-34]

33.   A method as in claim 11, wherein each of the frames is encrypted with a

different key. [pg. 6, ll. 33-34]

(VI) <u>GROUND(S) OF REJECTION TO BE REVIEWED ON APPEAL</u>

Claims 1, 4-16, 21-26 and 29-33 stand rejected under 35 U.S.C. §102 as allegedly

being anticipated by Iverson et al. (U.S. Patent No. 6,233,537, hereinafter "Iverson").

(VII) ARGUMENT

<u>Claims 1, 4-16, 21-26 and 29-33 are not anticipated under 35 U.S.C. §102 by</u>

<u>Iverson.</u>

For a reference to anticipate a claim, each element must be found, either expressly

or under principles of inherency, in the reference. Each element of the claims is not

found in Iverson. For example, Iverson fails to disclose *encrypting* each of a multiplicity

of data frames and then *decrypting* the data frames using keys as required by independent

claim 1. Similar comments apply to independent claims 11 and 12. In short, Iverson's

disclosure of *encoding/decoding* signals fails to disclose *encrypting/decrypting* data

frames as claimed.

Section 4 (pages 9-10) of the final Office Action mailed November 3, 2005 alleges

"wherein it is inherent by disclosing <u>encoded</u> multimedia bitstream comprises video

frames (col. 3, lines 5-7 and col. 6, lines 24-29) involves <u>encryption</u> that includes

generated random number generator that uses a seed value for each key frame (col. 9,

lines 59-66) (emphasis added)." Appellant respectfully disagrees with the apparent

allegation that Iverson's disclosure of <u>encoded</u> multimedia bitstream inherently discloses

<u>encrypting</u> each of a multiplicity of data frames and then <u>decrypting</u> the data frames using

keys as claimed.

Col. 2, line 56 to col. 3, line 4 of Iverson describes applying an encryption scheme

to file data to generate an encrypted file. This encrypted file would have to be decrypted

before being played. However, this portion of Iverson concludes by describing potential

disadvantages of an encryption/decryption scheme as "Requir[ing] a prohibitively

expensive amount of memory space" and "the delays resulting from performing the decryption procedure may be undesirable for applications that are designed to play audio/video sequences to simulate real-time sound and motion."

Iverson then contrasts the encryption/decryption procedures described in col. 2, line 56 to col. 3, line 4 with Iverson's invention which is directed to a scheme for "controlling the access that a user has to an <u>encoded</u> multimedia bitstream (emphasis added)." (See col. 3, lines 5-16). Accordingly, col. 3, lines 5-16 describes Iverson's system in which an *encoded* multimedia bitstream does not suffer from the disadvantages of the *encryption* procedure described in col. 2, line 56 to col. 3, line 4. <u>The explicit language described in col. 2, line 56 to col. 3, line 16 of Iverson itself explicitly describes that there is a difference between "encrypting" and "encoding."</u>

Similarly, Iverson's system includes a specially designed *decoder* (see col. 3, lines 5-16) to overcome the disadvantages of the *decryption* procedure (see col. 2, line 56 to col. 3, line 4 stating, *inter alia*, "delays resulting from performing the decryption procedure may be undesirable for applications...."). The explicit teachings of Iverson thus describes that there is a distinction between "decrypting" and "decoding."

Accordingly, encrypting a multiplicity of data frames with different keys is clearly not "inherent" in view of Iverson's teaching of encoding data. Iverson itself explicitly discloses that encrypting/decrypting and encoding/decoding are different concepts. The rationale of the final Office Action simply ignores these explicit teachings of Iverson contrasting encrypting/decrypting with encoding/decoding. Iverson explicitly contrasts encryption/decryption (described in col. 2, line 56 to col. 3, line 4) with

encoding/decoding (described in col. 3, lines 5-16). These portions of Iverson clearly

present encryption/decryption and encoding/decoding as <u>alternative</u> procedures. Iverson

teaches <u>away</u> from encryption/decryption procedures by discussing its disadvantages and

describing that his encoding/decoding resolves these disadvantages.

Page 10 of the final Office Action makes further reference to a Microsoft

computer dictionary with purports to define encryption as "encode (scramble)

information in such a way that it is unreadable to all but those individuals possessing the

key to the code."[1] It is significant to note that this definition indicates that information is

scrambled in such a way that it is *unreadable* to all but those individuals possessing the

key to the code. In other words, encryption requires some type of a cryptographic

algorithm and a cryptographic key to make information secret. At the very least, a

disclosure of a generic concept such as encoding (as in Iverson) fails to disclose the more

specific concept of encryption which specifically requires "in such a way that it is

unreadable to all but those individuals possessing the key to the code" (assuming that the

Microsoft computer dictionary definition is accepted).

In contrast to this definition, the frame data encoder described in Iverson processes

the current frame to generate encoded data using "standard video encoding techniques."

(See col. 6, lines 25-35). One skilled in the art would clearly realize that such standard

video encoding techniques do not involve any secrecy. No key is used to encrypt the data

and make it secret. Somebody who knows which technique was used to encode the data

frame code easily decode it without any additional knowledge.

---

[1] Appellant respectfully requests that the next Office Action provide a copy of this portion of the Microsoft computer dictionary.

In more detail, col. 6, lines 25-35 disclose the following standard video encoding

techniques used by frame-data encoder 302 of Iverson to generate encoded data (step

404): block-based motion estimation, block-based motion-compensated differencing,

block-based transformation, quantization of transform coefficients, run-length encoding

of quantized coefficients, and Huffman-type entropy encoding of run-length codes. None

of these standard video encoding techniques discloses encryption. For example, none of

these techniques involve any secrecy or use any key to encrypt data to make it secret.

Any allegation to the contrary would contradict col. 2, lines 56 to col. 3, line 16 of

Iverson which explicitly distinguishes an encrypting/decrypting procedure with an

encoding/decoding procedure.

Documents (1)-(3) attached to Section IX (Evidence Appendix) of this Brief

provide further evidence that encoding/decoding fails to disclose encryption/decryption.

Applicant submits that one of ordinary skill in the art would have understood these

distinctions between the terms encoding/decoding and encryption/decryption at the time

that the present application was filed. In particular, Document (1) states the following

(emphasis added, see pg. 5):

> "**'There is a big difference between encoding and
> encryption,'** he said. 'Encoding, when used in a software
> development context, typically means translating some concept
> to a digital form for use by the computer. For example, ASCII is
> an encoding scheme for the English alphabet and punctuation. In
> ASCII, the letter 'A' is encoded as the value 65, or 1000001
> binary. In fact, ASCII stands for 'American Standard Code for
> Information Interchange.' Thus, letters placed into this "code"
> are encoded in ASCII form. Similarly, in AutoCAD we have to
> translate things like geometry and attributes into a digital code to
> be interpreted by the computer and stored on the hard drive. DXF
> is one form of encoding. DWG is another. So, the concept of a

red line from 0,0 to 1,1 would be encoded as some series of binary numbers in the DWG. This is the manner in which we used 'encoding' in our original email with Evan's team.

'Encoding is a word in common usage in software engineering. **Unlike encryption, encoding does not imply any attempt to hide or obfuscate information.**"

Document (2) states, *inter alia*, the following (emphasis added, see pg. 1):

"Encrypting is the process of creating a system of secret writing based on a set of predetermined rules or symbols. Encoding and decoding are the processes of converting a message or information into and out of some code. **The difference between encryption/decryption and encoding/decoding is that encryption processes are meant to be secret.** One of the first uses of computers was in the 1940's by the British to break German secret codes."

Document (3), entitled "Encoding is Not Encryption" states, *inter alia*, the following (emphasis added, see pg. 1):

"**It's unfortunate that the words *encryption* and *encoding* tend to get used as synonyms. In cryptography they mean two different things.**
    In short:
•   Encryption = encipherment = make secret
•   Encoding = to convert format, not necessarily securely
    Encoding is not encryption. **Repeat, encoding is not encryption.** Strictly speaking, encryption is an encoding operation, but the term **encoding** is generally used in cryptography to mean that secrecy is not involved."

For the foregoing reasons, one of ordinary skill in the art would have understood

that Iverson's teaching of encoding/decoding fails to disclose encryption/decryption as

claimed since (1) Iverson itself explicitly contrasts encoding/decoding from

encryption/decryption, (2) the Microsoft dictionary definition cited by the final Office

Action distinguishes between the generic concept of encoding from that of the more

specific concept of encryption ("in such a way that it is unreadable to all but those individuals possessing the key to the code"), and (3) documents (1)-(3) of the Evidence Appendix of this Brief clearly disclose that encoding/decoding are distinguishable concepts from encryption/decryption. This understanding by one of ordinary skill in the art is consistent with the specification. (For example, see reference to "a unique public/private key pair" in page 9, lines 27-30 of the specification).

Since Iverson fails to disclose encrypting/decrypting as required by independent claims 1, 11 and 12, Iverson fails to anticipate these claims. The absence of any element of the claim from the cited reference negates anticipation. See, e.g., *Structural Rubber Prods. Co. v. Park Rubber Co.*, 749 F.2d 707, 715 (Fed. Cir. 1984). Anticipation is not shown even if the differences between the claims and the prior art reference are insubstantial and the missing elements could be supplied by the knowledge of one skilled in the art. See, e.g., *Structural Rubber Prods.*, 749 F.2d at 716-17.

Moreover, Iverson further fails to disclose the following limitations of independent claim 1 (similar comments apply to independent claims 11 and 12):

> "f)   passing a control message, for modifying and controlling the availability of keys, in the control field to the secure module at a selected one or more users, and
> g)   at the secure module of the or each selected user, in response to the said control message, controlling the availability of keys generated from the said seed value, thereby controlling access by the users to the said data."

Col. 9, lines 55-67[2] disclose a random-number generator that uses a seed value known to both the encoder and decoder. The random number generated from this

---

[2] Immediately preceding col. 9, lines 50-54 of Iverson are specifically identified in the final Office Action.

random-number generator using the seed value can then be input into a hash function

along with an access word. The result of that hash function (i.e., the hash result) using

the access word and random number can then be compared to a lock word to determine

whether to decode data. However, there is no disclosure in this portion of Iverson (or any

other portion) of controlling the availability of decryption keys generated from a seed

value. The random number generated using the seed value is simply used as an input to a

hash function.

Independent claim 12 further requires "encrypting the plurality of frames with

different respective keys." Similarly, dependent claims 32 and 33 (which depend from

independent claims 1 and 11, respectively) further require "wherein each of the frames is

encrypted with a different key." Iverson fails to disclose different keys.

With respect claims 12 and 32-33, page 9 of the final Office Action makes explicit

reference to col. 8, lines 55-58. This portion of Iverson states "The computationally

simple method employed to hash the access word has several advantages. First, it

produces a lock word which differs for each encoded frame, which makes the key frame

lock-word field much more difficult to detect." However, the lock word is only encoded

into the frame header. That is, it prefixes the frame data. The lock word is not used as an

operator on the frame data itself. The lock-word of Iverson is not used as an operator to

encrypt the data frame. Rather, the frame data is in the clear.

Col. 8, lines 39-54 of Iverson provides further evidence that the lock-word of

Iverson has nothing to do with an encryption key. This portion of Iverson describes the

purpose of the lock-word and what would happen if someone turned-off the key-frame

lock-word protection. What is clear is that they would not have to decrypt any of the frame data. Rather, they would have to reconstruct the headers and adjust the frame size.

Dependent claim 6 further requires the control message including a user identity field identifying a selected user or group of users so that the control message is acted on only by the user or group of users identified by the user identity field. Dependent claim 7 further requires the control message including a stop flag for stopping the generation of keys, and dependent claims 29-31 similarly require the control message causing a module to cease releasing keys. Dependent claim 9 further requires the control message including a contact sender flag. Iverson fails to disclose each of these specific types of field or flag. For example, frame-header generator 308 of Iverson generates a frame header for a particular key frame. The frame-header merely includes a lock-word bit, a 32-bit lock-word and a checksum value. (See col. 7, lines 1-9). The checksum value may be substituted with a seed value as described in col. 9, lines 55-67. While these portions of Iverson discloses a lock-word bit, lock-word and checksum (or seed) value, there is no teaching or suggestion of a user identity field as required by claim 6, a stop flag as required by claim 7, a field for ceasing releasing of keys as required by claims 29-31 and/or a contact sender flag as required by claim 9.

Dependent claims 23-24 require a primary seed value and secondary seed values. Col. 5, lines 57-58 (specifically identified by the final Office Action as disclosing the subject matter of claim 23) merely states "Receiver 210 may be any suitable means for receiving the digital signals transmitted by transmitter 118 of encoding system 100." This clearly does not disclose a primary seed value or secondary seed values as required

by dependent claim 23, let alone generating a secondary seed value from the primary

seed value by modifying the primary seed value with a source identity value as further

required by dependent claim 24. Col. 9, lines 55-64 (also specifically identified by the

final Office Action) merely discloses a random-number generator that uses a seed value

known to both the encoder and decoder. There is no teaching or suggestion of a primary

seed value and secondary seed values.

Appellant thus requests that the rejection of claims 1, 4-26 and 29-33 under 35
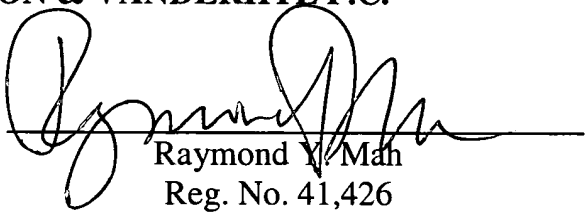
U.S.C. §102 be reversed.

CONCLUSION

In conclusion it is believed that the application is in clear condition for allowance;

therefore, early reversal of the Final Rejection and passage of the subject application to

issue are earnestly solicited.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: Raymond Y. Mah
Reg. No. 41,426

RYM:sl
901 North Glebe Road, 11th Floor
Arlington, VA 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100

26

<div align="center">

(VIII)   <u>CLAIMS APPENDIX</u>

</div>

1.   A method of distributing digitally encoded data, comprising:

a)   dividing said data into a multiplicity of frames,

b)   encrypting said frames,

c)   distributing multiple copies of the said data frames to a multiplicity of users, each frame being distributed with a control field,

d)   communicating a seed value for key generation to respective secure modules located at each of the multiplicity of users,

e)   decrypting the data frames at respective users using keys derived from the seed value communicated to the secure module, the secure module being arranged to enable decryption of a respective frame only when said control field has been passed to the secure module,

f)   passing a control message, for modifying and controlling the availability of keys, in the control field to the secure module at a selected one or more users, and

g)   at the secure module of the or each selected user, in response to the said control message, controlling the availability of keys generated from the said seed value, thereby controlling access by the users to the said data.

4.   A method according to claim 1, in which each data frame includes a frame identity field, and each key generated by the secure module is specific to one frame identified by the said field.

<div align="center">

27

</div>

5.   A method according to claim 1, in which the step of distributing multiple copies of the said data comprises multicasting packets of data via a communications network to the plurality of users.

6.   A method according to claim 1, in which the control message is distributed with a data frame to the multiplicity of users, a user identity field identifying a selected user or group of users is included in the control message, and the control message is acted on only by the user or group of users identified by the said user identity field.

7.   A method according to claim 1, in which the control message includes a stop flag, and in response to the stop flag the generation of keys at the or each selected user is stopped.

8.   A method according to claim 1, including returning a response signal from the secure module to the source of the control message.

9.   A method according to claim 8, in which the control message includes a contact sender flag, and the step of returning a response signal from the secure module is carried out when the contact sender flag is set.

10.   A method according to claim 8, including transmitting a further control message to the user on receipt of the said response signal.

11.   A method of operating a customer terminal in a data communications system, the method comprising:

a)   receiving at the customer terminal a multiplicity of encrypted data frames, each with a control field;

b)   receiving at the customer terminal a seed value for key generation;

c)   passing the said seed value for key generation to a secure module located at the customer terminal;

d)   generating in the secure module using the seed value keys for the decryption of data frames;

e)   decrypting using the said keys only those respective data frames for which a control field has been received;

f)   passing to the said secure module a control message received in the control field; and

g)   in response to the said control message, controlling the availability of keys generated using the said seed value and thereby controlling access by the user of the customer terminal to data received at the customer terminal.

12.   A data communications system comprising:

a)   a remote data source arranged to output a plurality of frames;

b)   encryption means for encrypting the plurality of frames with different respective keys;

c)   a communications channel arranged to distribute multiple copies of the

encrypted data frames, each with a control field;

d)   a multiplicity of customer terminals arranged to receive from the

communications channel respective copies of the encrypted data frames with the control

fields;

e)   a key generator located at a customer terminal and programmed to generate

from a seed value keys for use in decrypting data frames;

f)   key control means connected to the key generator, the key control means

comprising:

an interface for receiving the control fields ; and

control means arranged to only release keys for decrypting those respective

frames for which a control field is received and being arranged to, in response to

the said control messages in the control fields, control the availability to the user

of keys generated from the seed value; and

g)   decryption means connected to the key generator and arrange to decrypt the

data frames received at the customer terminal from the communications channel.

13.   A data communications system according to claim 12, in which the

communications channel is a packet-switched data network.

14.   A customer terminal for use in a method according to claim 1, the customer

terminal comprising:

a)   a data interface for connection to a data communications channel;

b)   a key generator programmed to generate from a seed value keys for use in decrypting data frames:

c)   decryption means connected to the data interface and the key generator and arranged to decrypt data frames received via the data interface; and

d)   key control means connected to the key generator, the key control means comprising:

an interface for receiving control fields; and

control means arrange to only release keys for decrypting those respective data frames received with a control field;

the control means being arranged to in response to control messages in the control fields, control the availability to the user keys generated from the seed value.

15.   A data server for use in method according to claim 1, the data server comprising:

a)   a data interface for connection to a data communications channel;

b)   means for outputting encrypted data frames with control fields via the data interface onto the communications channel for receipt by a multiplicity of customer terminals;

c)   means for outputting the control fields having control messages onto a data communications channel for controlling the operation of key generators at customer terminals.

31

16.   A method according to claim 1, including generating keys from the seed value by iterated operations on the seed value by selected ones of a plurality of predetermined functions.

21.   A method according to claim 1, including applying different characteristic variations to data decrypted at different respective customer terminals.

22.   A method according to claim 1, including a plurality of remote data sources, each outputting a respective plurality of frames.

23.   A method according to claim 22, in which the customer terminal receives a primary seed value common to different respective data streams from the plurality of data sources, and derives from the common primary key a plurality of different respective secondary seed values for decrypting frames from different respective data sources.

24.   A method according to claim 23, in which data received from different data sources includes different respective source identity values, and the respective secondary seed value is generated from the primary seed value by modifying the primary seed value with the source identity value.

25.   A method according to claim 1, in which each data frame includes a frame type field.

26.   A method according to claim 25, including storing a receipt including data from the frame type field.

29.   A method as in claim 1, wherein the control message received by the secure module causes the secure module to cease releasing keys.

30.   A method as in claim 11, wherein the control message received by the secure module causes the secure module to cease releasing keys.

31.   A data communications system as in claim 12, wherein the control message received by the key control means causes the key control means to cease releasing keys.

32.   A method as in claim 1, wherein each of the frames is encrypted with a different key.

33.   A method as in claim 11, wherein each of the frames is encrypted with a different key.

(IX) <u>EVIDENCE APPENDIX</u>

(1) Martyn Day, Ed., *AutoCad 2004 DWG:Not Encrypted, Honest!*, CAD Digest,

pgs. 1-7 (July 17, 2003)

(http://www.caddigest.com/subjects/autocad/select/day_encryption_2004.htm)

(2) *Wafer Encryption (Candy Grams)*, University of Florida Civil Engineering,

pgs. 1-5.

(http://www.ce.ufl.edu/activities/wafer/wepins.html)

(3) *Encoding is Not Encryption*, DI Management Services Pty Limited, pgs. 1-2

(May 13, 2006 = last update).

(http://www.di-mgt.com.au/encode_encrypt.html)

(1) Martyn Day, Ed., *AutoCad 2004 DWG:Not Encrypted, Honest!*, CAD

Digest, pgs. 1-7 (July 17, 2003)

(http://www.caddigest.com/subjects/autocad/select/day_encryption_2004.htm)

CAD DIGEST

The reading room for computer aided design

home - about - advertise

Search: [_____] **(GO)**

Navigation

Partners

TenLinks.com

CADdepot.com

upFront.eZine

AutoCAD   *Feature*

AutoCAD 2004 DWG: Not Encrypted, Honest!

By Martyn Day, editor, CADserver, July 17, 2003

See Also
· Autodesk website
· OpenDWG Alliance website
· TopTen AutoCAD 2004 Sites - by TenLinks
· Autodesk Directory - by TenLinks.com
· Autodesk Reading Room - by CADdigest.com

On April 1st, 2003, the OpenDWG Alliance publicly announced that its users were having trouble reverse-engineering the new AutoCAD file format. The reason for the problem, the group claimed in its press release, is that Autodesk had used complex compression, together with data encryption techniques: "the inclusion of data encryption and compression schemes within the new file format has created serious challenges to DWG data interoperability."

This is a serious accusation. What's most significant is not the mention of file format compression (most file formats are compressed) but the allegation that the DWG file is actually "encrypted." All sorts of negative and nefarious connotations could arise from the suggestion that Autodesk has adopted encryption; namely, that the company is trying to lock in its users and keep out the competition, since opening a file would require every user to possess a valid copy of Autodesk software.

OpenDWG's claims

For those not up on the wonderful world of data interoperability, the OpenDWG Alliance was born out of the need of Autodesk's competitors to reverse-engineer AutoCAD's DWG file format. To become a member, you have to tell the OpenDWG Alliance all you know about the DWG format, pay a yearly fee and in return receive the DWG libraries that its programmers generate. The OpenDWG's goal is to make DWG an open standard "usable by all programs that need to access valuable DWG data." Needless to say, with proprietary file formats being seen as a competitive advantage, Autodesk isn't a member (the de facto standard, DWG files are estimated to number some 3 billion). In my talks with Autodeskers though, they wryly point out that they would join the Alliance if they were to receive libraries of all the other member's proprietary CAD products in return (those by Bentley, EDS, PTC, Graphisoft, Nemetschek, ESRI. and CADKEY to name a few).

After reading the Alliance's release, I spoke with Evan Yares, the group's president and executive director. Yares was convinced that Autodesk had chosen to encrypt the DWG file, despite assurances from Autodesk representatives that they had not. He sent me a DWG file with a corresponding Hex dump, with an explanation of how this demonstrated Autodesk's encryption. Unfortunately, not being a programmer, the evidence was somewhat hard for me to digest.

In an issue of the upFront.eZine newsletter, the OpenDWG Alliance made specific technical claims concerning the compression algorithm: "Rather than having a single compression type, each object type appears to have its own individual algorithm, with a large number of special cases. Object compression is controlled by a 32-bit flag, which provides for billions of possible permutations. OpenDWG has reverse-engineered the compression algorithms for some objects, but substantial work remains to be done." And on the encryption issue, the group stated: "Both the file and section headers are encrypted, but in different manners from each other. While OpenDWG has been able to determine the algorithm used for both, it has not been able to determine if the encryption keys used to scramble the data will remain static, will change in each point release of AutoCAD, or will ultimately be changed dynamically under program control." (See DWG 2004 - Tougher Than Thought.)

Surely, faced with such a specific and detailed accusation, Autodesk would respond? But the company appeared to hold back. Carl Bass, executive vice president of Autodesk's Design Solutions Division, told me that the OpenDWG Alliance claims were "total nonsense and hysteria." It seemed that Autodesk was going to wait it out; in time, Bass expected, the Alliance would realize there was no such encryption and make a public announcement or apology to that effect.

The plot thickens

Meanwhile, Montreal-based viewing and mark-up developer Cimmetry announced that it had support for the AutoCAD 2004 file format in its forthcoming update to AutoVue 17 - the same DWG format that the OpenDWG Alliance claimed was encrypted. Cimmetry had reverse-engineered the file format without any technology input from Autodesk. While this could not be taken as a sign that 2004 DWG was not encrypted (Cimmetry has reverse-engineered both non-encrypted and encrypted file formats in the past), it was a sign that the format was at least decipherable by an outside organization. In his CADwire.net commentary, Evan Yares, this time in his role as industry analyst, chose to doubt the validity of Cimmetry's claim: "Either Cimmetry told the truth in their press release, or they lied. I've seen no evidence that Cimmetry told the truth." I assumed that this comment reflected his role within the OpenDWG and the extra pressure that group must have felt on hearing that someone else had cracked 2004 DWG.

I managed to get a copy of the updated 2004 DWG AutoVue to test it for myself and then showed Yares that Cimmetry had indeed cracked the 2004 format. Yares then honorably posted a retraction: "Cimmetry can fairly claim bragging rights to having delivered the first third-party viewer with AutoCAD 2004 support. I definitely owe them this recognition, and my congratulations." It is true to say though that reverse-engineering to read a file is only halfway there, the OpenDWG would have to be able to write 2004 DWGs as well.

All this time, Autodesk had remained silent. There were times past when I would have expected Autodesk to have fired off a few lawyers' letters, or at least produced a qualified rebuttal. When I asked to interview Carl Bass on the subject of encryption, he agreed, and we

were joined by John Sanders, executive VP of Design Solutions
Division, and Mark Strassman, director of marketing for AutoCAD.

Autodesk responds

I first asked for Autodesk's reaction to OpenDWG's claim that 2004
DWG was encrypted. Bass replied, "The whole thing is actually pretty
complex and I think people like Evan (Yares) have muddled the facts.
There is an element of truth in many of the things they say but the gist
of their argument is not correct. We changed the DWG file format for
the customer's benefit. A lot of this benefit was around network
performance and compression. We made no secret of the fact that we
were compressing files and not very differently from something like a
Zip file. We didn't use the same algorithm as Zip, but we did use a
relatively standard and well-known compression method. As is often
the case in computer problems, there is a trade-off between size and
speed. Because of the interactive nature of a program like AutoCAD
vs. zipping/unzipping files for email or archiving, we selected an
algorithm that was optimized for performance."

Strassman added, "The smaller file size is one of the big features of
2004. One of the things Evan said is that we use different compression
types for individual features and stuff like that, which is just false. We
use a standard compression library throughout the DWG file. It's
standard compression, there's no encryption. Compression just makes
the file smaller."

Bass said that files were growing larger and larger and sharing them
was beginning to overtax networks. "Now, if users only have to move 3
to 5 megabytes instead of 10, that's obviously better for everyone on
the network. Just as we do when we send people big files, we often
compress them, that's why people send JPEGs around instead of
TIFFs and another reason why we came up with DWF. It's all about
moving the information around more efficiently. We surveyed the
customers and a vast majority run AutoCAD, or AutoCAD-based
products, on a network, and this drastically improved the Open, Close
and Save performance across a network. Compression is about better
performance over networks - pure and simple. But if what we were
doing was only that, you could decompress it at the other end and look
at a file that was a 2002 file. We hadn't changed the file format in a
while, because to do that means you have re-architect it. On the first
release you do great. The second one gets a little bit messy, the third
one gets pretty crusty and then you need to clean it up. And it's worth
knowing where you are going for the next several releases, when you
don't want to change file format. That's always been a barrier for users.
So it makes sense to put in place an architecture that allows for that
kind of extensibility."

John Sanders added, "Hopefully, we have a foundation for the next
couple of releases so we won't need to change the format."

While that sounds a reasonable explanation, Autodesk changes its file
format, on average, every 3 releases, which doesn't seem to be very
forward looking. I asked Bass why this was the case when competitors
such as Bentley managed to keep the same format for 15 years.

"If you go into Microsoft Word and you pull down the Save As menu,
there are about 7 different file formats to choose from. You change
formats because you have to add functionality. I really don't know
enough to comment on Bentley."

Strassman explained, "A lot of the changes in the past have been

considered minor because we were just stuffing more information into the DWG file format. This time, we wanted to make it so the file would really have a future, so we could eventually add new features to AutoCAD without changing the file format."

But wasn't that what the Object ARX programming language was developed for? "That's what ARX did from a code standpoint, but not from a data standpoint," Bass replied. "We've just made a much more flexible way to add data for us and for our third-party developers. They have a better mechanism to get at the data. So that's what that was all about, it's all geared toward customer benefit. That's why we changed the file."

Strassman expanded on this point: "We also did a bunch of other things ato make it easier to recover files, to make it easier to see when a file is corrupted. We added all sorts of things for the functionality of the user, which will hopefully allow us to avoid changing the file format significantly in the future."

"Mark brought up an important point," added Bass. "People have always wanted a reliable 'recover' command. AutoCAD always done a reasonably good job but it has never been 100% perfect in that area. But data recovery is a huge user request, especially because people get corrupt DWGs written by third-party DWG libraries - which is ironic in the context of this conversation. The problem is compounded when an architect, for example, creates a file, sends it to a consulting engineer and they apparently corrupt the file and then they want to be able to recover it and get the information back. So we have put in more mechanisms to make sure we could actually provide a recover in more circumstances.

Encryption vs. encoding

On broaching the encryption issue, I reiterated the OpenDWG's specific arguments concerning their findings, namely that the file headers are scrambled with a 128-byte magic number. This statement incited rebukes all around from my interviewees.

Bass and Saunders both chimed in, "Wrong, wrong, wrong!", while Strassman added, "There is no magic number."

"They will find out later," said Bass. "That's why I haven't been particularly interested in responding because the competitors just look incompetent."

What about the other OpenDWG allegation - that the sub-headers in the file are encrypted with a 4-bit key? "There is no encrypting," Bass replied. "None. There are no keys, they are wrong. I'm more than happy to make a statement about it not being encrypted, except for the password protection. We actually believe that customers should have total control over their data. As an example, we had a choice when it came to putting encryption for the password protection in the file, whether or not to have a back door to access the data and we decided not to. There is nothing special that we can do to that file that a user can't do. It's totally down to user control. We believe fundamentally that users have a right to control their data."

The ownership of data is an interesting point. Should users only have access to their data though an Autodesk product? Bass replied, "If they created it in a DWG file format that's not from an Autodesk product, I don't think we are that involved in this question, right? If they created the DWG in MicroStation, we would have no real involvement in that

conversation, they just happen to have chosen our format but we have no obligation, one way or another. The relationship is entirely between that user and Bentley."

If a non-Autodesk customer was given a DWG file to work with, does Autodesk believe that person should buy a copy of its CAD software to open it? "No, it's up to them to look along the axis of price, features and fidelity," answered Bass. "I think that person is someone we don't have an obligation to. Are we interested in them becoming a customer? - that's a different question than whether we have an obligation to provide them with free software or with certain functionality." Bass added that he was actively considering allowing an independent third-party to reverse- engineer the DWG file format to adjudicate on the issue. He said he wouldn't pay for it to be done but the doubters could. To offer DWG up for independent analysis would be a foolish thing to do if 2004 were indeed encrypted. I took this alone to be a sign that Bass was willing to put his head on the block to state that AutoCAD 2004 DWG was not encrypted.

So, on the question of encryption, an emphatic "No" from Autodesk - along with some bruising comments on the capabilities of the OpenDWG Alliance. Speaking of which, I've seen correspondence between Autodesk and the OpenDWG, in which the former admitted there was some "simple encoding" in the new DWG. I asked Mark Strassman what this meant.

"There is a big difference between encoding and encryption," he said. "Encoding, when used in a software development context, typically means translating some concept to a digital form for use by the computer. For example, ASCII is an encoding scheme for the English alphabet and punctuation. In ASCII, the letter 'A' is encoded as the value 65, or 1000001 binary. In fact, ASCII stands for 'American Standard Code for Information Interchange.' Thus, letters placed into this "code" are encoded in ASCII form. Similarly, in AutoCAD we have to translate things like geometry and attributes into a digital code to be interpreted by the computer and stored on the hard drive. DXF is one form of encoding. DWG is another. So, the concept of a red line from 0,0 to 1,1 would be encoded as some series of binary numbers in the DWG. This is the manner in which we used 'encoding' in our original email with Evan's team.

"Encoding is a word in common usage in software engineering. Unlike encryption, encoding does not imply any attempt to hide or obfuscate information. Because laypersons occasionally confuse the usage of 'encoding' and 'encryption' we have stopped using the term encoding when referring to the DWG. The only encryption in the AutoCAD 2004 DWG is file password protection, which is totally under the control of the user and is there to allow for secure transmission of drawings solely at the user's discretion."

OpenDWG support

On May 16th, the OpenDWG Alliance announced official support for the 2004 variant of the AutoCAD DWG file format. In the press release Evan Yares stated that, "Although we had no significant concerns about being able to implement support for the AutoCAD 2004 DWG file format, there were enough variables that the task was not trivial." This statement, to me, is a bit rich since those "insignificant concerns" had generated an attacking press release only the month previous. The release went on to claim that, "In AutoCAD 2004 DWG, a comprehensive compression algorithm is applied to almost all data structures, and the file and section headers are encrypted using a magic-number/XOR algorithm." A cautionary note to users added that,

"Despite the fact that we support the format, users should continue to be cautious about using AutoCAD 2004 DWG files for projects which require long-term data access as the format does contain encryption."

Again, a series of specific claims, although the Alliance seems to have dropped an earlier allegation that there were billions of magic number permutations and that the encryption could be changed on the fly without introducing new builds of AutoCAD.

Conclusion

The AutoCAD 2004 DWG encryption debate is a very complex one to follow, as relates to both an understanding of the technology itself and the semantics of the arguments. I have to believe Carl Bass when he says that Autodesk has not encrypted the file - not in the classic definition of running an algorithm over the DWG to hide its contents from all outsiders. If that had been the case then it was a failure because Cimmetry announced support within a month of AutoCAD 2004's shipping and it only took the OpenDWG Alliance a month beyond that. Besides, Autodesk is savvy enough to foresee the kind of bad press that would result from doing something as outrageous as encrypting the basic DWG file and, in effect, trapping their customers.

That said, Autodesk has done a major amount of work to the AutoCAD file format and data compression is, in some way, data obfuscation, where data is reduced in size via a formula (algorithm) and reconstituted on loading. As compression is standard across the industry, one could hardly point a wagging finger at Autodesk. I have it on good authority, however, that the compression used in AutoCAD 2004 is very complex and doesn't appear to give DWG any greater ratio of compression than PKZIP provides. So why adopt it? I haven't had a sufficient explanation from Autodesk.

Although Autodesk obviously hasn't overtly encrypted the 2004 DWG, certainly it isn't in the company's interest to make the reverse-engineering process any easier for its competitors. Autodesk has made code changes to AutoCAD LT in the past to "dissuade" application developers from coming up with applications for AutoCAD LT. It may well be that this was taken into consideration when the new DWG was being devised; with all the changes, improvements, and semantics it's difficult to see the big picture.

If anything, Autodesk should be more worried about *why* people were so willing to believe that it had overtly encrypted the file format. Indeed, on my travels it appears that the general perception is that Autodesk has used encryption; "It sounds like something Autodesk would do" being a typical response. In light of such widespread negativity, perhaps Autodesk should respond with more openness to its customers and to the industry at large. Autodesk's negativity towards Adobe's PDF format makes one think that format definition and control of those formats really does matter to Autodesk.

One of the biggest issues in the CAD world is interoperability, the battle between proprietary systems and open formats. Nearly all CAD systems are in some way proprietary because they are devised and controlled by the company that originated them. As a customer of these software firms you own the information that is stored in their "wrapper" (file format) but do not always have an independent way of gaining access to that information. The OpenDWG Alliance claims it is acting for the good of open systems. But it's worth noting that the group's existence is funded by competitors to AutoCAD and its DWG format. I think this is an awkward position to defend. As for Autodesk, it tends to rest its "open format" laurels on DXF, developed in the 1980s

to solve the company's own problems of transferring DWGs between incompatible operating systems.

There are no real saints here, on either side of the divide.

About the Author

Martyn Day is group editor of *MCAD Magazine* and *AEC Magazine*. For more information, visit the CADserver website.

Related Articles

(2) *Wafer Encryption (Candy Grams)*, University of Florida Civil

Engineering, pgs. 1-5.

(http://www.ce.ufl.edu/activities/wafer/wepins.html)

WAFER ENCRYPTION

(Candy Grams)

INSTRUCTOR

OVERVIEW OF THIS PROJECT - WHAT TO EXPECT

This project has the students design, remember, and test a process that allows quick and easy encoding and decoding of a message using SweetTart candies. With a limited number of candies and specified message lengths that include both letters and numbers, this project challenges students to create efficient but easily remembered processes for encoding and decoding. The students are not allowed to use any written material when they encode and decode.

In addition, the students must devise a means to ensure that their messages will not get "garbled" during transmission. Student teams will split and go into different rooms when they are ready to test. Using only paper, the binding of a book, and/or the original plastic wraps from the candies, students need to design a system to keep their message parts together while you or another neutral party delivers messages between rooms.

SPECIFIC PURPOSE OF THIS PROJECT

This project gives students a chance to think about processes for information transmission or what is known as telemetry. Students might think about and create efficient and simple processes. Their processes must be efficient since they will have only a limited amount of "wafers" to create a "long" message. Their processes must be simple since all members of the team will need to remember the processes. Teamwork is essential since all members of each team will participate and be relied upon by the other members of the team. Cooperation is paramount!

There are several related engineering and mathematical concepts to this project. Specifically, encryption methods, encoding and decoding, digital processes, and number systems. Encrypting is the process of creating a system of secret writing based on a set of predetermined rules or symbols. Encoding and decoding are the processes of converting a message or information into and out of some code. The difference between encryption/decryption and encoding/decoding is that encryption processes are meant to be secret. One of the first uses of computers was in the 1940's by the British to break German secret codes.

Digital processes are those that use groups of electronic bits that are either on or off. Digital processes are based on a binary number system. Most students learn about different number systems in algebra (base two, base eight, base ten and base sixteen). This project gives them an opportunity to directly use other types of number systems (possibly base two, base three, base six, or some others) as a basis for their coding of letters and numbers, similar what digital computers use.

THE REAL-WORLD PROBLEM RELATED TO THIS PROJECT

Transmitting information, whether data or simple messages, is one important aspect of our information society. There is a multitude of different ways to send/receive, view, and store all this data. As examples of encoding/decoding consider the following:

- Satellites must convert their sensor's information into a form that can be sent back to earth bound scientists.
- Music studios take live music and convert it into a form that makes replaying sound as if it were live.
- Computers covert what you type at a keyboard into an electronic form that can be saved to magnetic disks.

Each of these examples uses a different process for converting some information, whether pictures, sound, or letters and numbers, into an electronic form that can be easily manipulated. Without encoding and decoding processes our information society would be without information. This project gives your students a chance to develop a new, innovative

way of communicating using an unusual material, SweetTarts.

An important aside to this project is cooperation and mutual support of all team members. Not only do all team members need to participate in the design of an encryption scheme, but each must also remember it and be relied upon to use it. Teamwork is not only important on the playing field; it is also important in the real world, particularly in design.

SPACE AND TIME REQUIRED

This project is an indoor one that requires several tabletops and possibly two classrooms. A flat nine square foot size work space per team is adequate. It is best to separate teams as much as possible, so each team can independently develop a coding scheme.

Testing is best done with two rooms, so you can divide each team in half and place them in different rooms. This eliminates any possibility of communication between team halves other than via the SweetTart message.

This project from introduction to testing requires at least four hours, which you can break into several blocks. There are several variations and extensions that you can add to this project, which will take additional time. Overall, it takes about two hours to design the process for encoding, decoding, and transmitting. You should allow the students about an hour to memorize and practice their process and an hour for testing.

MATERIALS AND EQUIPMENT NEEDED

The primary required material for this project is a sufficient amount of SweetTart candies. You can find these in most grocery stores in eight ounce bags. Each eight-ounce bag contains thirty-four rolls of candies. Each roll has twelve candies of six random colors. Each half team requires a total of fifteen twelve packs, so they can simultaneously be coding messages to each other. You will also need a digital watch for timing the time to encode and decode, and you will need a pad of paper to record the team times.

Students will also need pencils or pens and paper, so they can write down portions of messages and such. Paper cups are also useful to hold spare SweetTarts. You will have to remind students to keep track of their SweetTarts. We suggest that after students encode the messages they place the spare SweetTarts in a cup so the other half of the team will have them available for the next message that they will have to code.

You will need a pencil or pen and a piece of paper in order to record the encoding and decoding times of the different teams. A digital stop watch will help you record the times; however, a wall clock will do if a digital stop watch is unavailable. You might be able to "borrow" a stop watch from another teacher or even from one of your students in the class.

SUGGESTIONS REGARDING STUDENT TEAMS

Student teams should be made up of three to four students. The best size is four since the team can split evenly. Three students in a team are better than two since a primary focus of this project is to teach them about cooperative processes. Teams made up of more than four students tend to offer a greater chance of ignoring a student. You can observe the participation of each team member and divide the team such that students who are not taking part are "forced" to do so.

You should encourage the students to test their process before they split up. They can easily test by writing on a piece of paper a sequence of SweetTart colors (i. e., Red - Red - Green - Orange, to signify four SweetTarts in that sequence), just as they would actually appear. Each student in a group can encode a short message to every other student in the team. Thus, with a team made up of four students, each student will encode three messages to send and decode three messages received.

After students have practiced their encryption process, they can split into two groups and test their process for arranging the actual SweetTarts such that they may be sent to another room. Students will overlook this aspect of the project if you

do not remind them of it, and a good encryption process is doomed to fail if a message cannot be sent.

PREPARING FOR THIS PROJECT - WHAT TO DO IN ADVANCE

You can calculate the quantities of required materials based on the number of student teams. Each team should have either three or four students, and each team will require 360 SweetTarts. Each team should have twelve paper cups in which they can organize the different colored candies.

You can find the SweetTarts in most grocery stores. Each bag costs anywhere between $1.00 and $1.50. If you do not have paper cups readily available, then paper towels will also work.

It is best to make up appropriate length messages prior to class since you might not have enough time while students are designing. Your messages should be as close as possible to the message length limit of fifty characters (including spaces). Shorter messages will not fully exercise and challenge the student's processes. Adding numbers and words that use infrequent letters (i. e., q and z) will also challenge their schemes. The best messages to challenge the students will be those that do not have a large number of repeating letters and numbers and that cannot be deciphered from the context of the message. It is also a good idea to add some humor to the messages.

IMPORTANT STEPS IN MANAGING THIS PROJECT

We recommend the following approach:

1. Divide the class up into teams of three or four students.
2. Distribute the project description to the students and present the project. Students will need about twenty to thirty minutes of explanation, which includes time for them to ask questions.
3. Let the teams discuss and develop the principle basis behind their coding scheme. Teams will probably require from thirty to forty minutes for this task. While they are doing this you will want to take a few minutes with each team to discuss the principles behind their process. Some teams might need some help, while others will surprise you with their innovations.
4. Let the students take another hour for each member of the team to memorize their scheme and practice sending and receiving messages. Again, during this time, check up on each team so see if they have encountered any unforeseen problems. By showing your concern and helping students get over any difficult parts, all participants are ensured of having a rewarding experience.
5. Distribute the SweetTarts and paper cups to each team. Make the students aware that the SweetTarts must be split in half for each part of the team. Remind them that they also need to design a transmission process to get the messages between rooms. They will need about thirty to forty minutes to organize their SweetTarts and develop and test a transmission process.
6. Divide the teams in half and have each half of a team take their allotted number of SweetTarts, fifteen - twelve packs of random colors. Once all team halves are in separate classrooms, give each classroom a message to code. As the teams finish encoding their messages, they can bring them up to you at a central location and you can write down the encoding times.
7. After all teams have encoded a message, distribute the encoded messages for decoding. Again, after each team half is finished they can bring the decoded message to you and you can record the time to decode.
8. If time allows, you might want to have them encode and decode a second set of messages. (See the next section for possible variations of the testing process.)
9. When you are finished sending messages, calculate the scores for each team. During this processes, you might want to give each team a chance to comment on their results. For instance, they might have mixed up several letters or taken a long time to encode. This is a good time for you to add your observations.
10. Spend several minutes summarizing the results and providing closure after each team has commented on their design.

TESTING THE DESIGNS

You should have the teams split up and in different rooms for testing. None of the teams should have any pre-written notes on any paper, but teams may use paper and pencils during encoding and decoding.

It is impossible to be in two rooms at the same time; however, if you should choose a central location (e. g., a desk or chair in the hall between two rooms) to use as a base. You will need to inform all teams that they must bring their messages to that location where you will record their times. If this is unworkable, a desk in the front of one of the rooms will also work well.

Encoding

To be fair, you will have to have each team encode the same message; therefore, you will need to distribute that message to all teams at the same time. The easiest way is to pass out folded pieces of paper with the message to each team. Start timing as soon as you begin to reveal the message. Since you cannot be in two rooms at the same time, you might want to start one room one minute behind the other. After you have distributed the messages, you can take your place at the base and record each team's time when they bring a message to you. You can label each message with a piece of tape with each team's name, so you can distribute the messages to the other half of the teams when it is time to decode. You will also need to ensure the secrecy of messages so that the other half of the class does not accidentally them.

Decoding

After all the teams have brought their encoded messages to you and you have recorded their encoding times, you can distribute the appropriate encoded message to each team for decoding. Again, each team should only start with a blank piece of paper and pencil. Fairly distributing the messages to each team and timing them is not easy. The best way to keep the students under control is to have them all get their messages from you and not start until all teams have their messages. Since the team halves will be in different rooms, you should again use the one or two minute delay between rooms.

For those teams that finish before others, they should bring their encoded/decoded messages to you at your base and then quietly return to their seats and wait until all teams are finished. If any teams disrupting, you can assess penalty time to those teams.

Variations

There are a number of fun variations to this project that you can undertake if you have time. They are only limited by your imagination and time available. One interesting one is to give each team a message that you have encoded using your own scheme. What they must do is decipher it and break your code. The team that does this with the fewest mistakes and in the least amount of time achieves the best score. For students to accomplish this task in a reasonable amount of time without a computer, you will need to give them a starting point or "hint". The best hint is to give them several letters and numbers from your process. They will then use their imagination and the message itself to fill in the rest of your scheme. This is a hard task!

IMPORTANT PRINCIPLES INVOLVED

Humans first developed encoding schemes when ancient civilizations began drawing symbols on cave walls thousands of years ago. This simple form of communicating gradually developed into hieroglyphics and later into alphabets and number systems. Books evolved as a means of saving and transporting information and now in the twentieth century we have digital tapes, floppy disks, and CD's, to name a few means, to store and transport information.

Possible encryption methods can involve many different schemes. Students will surprise you with their ingenuity and creativeness. The most straightforward and easiest methods involve setting up a based number system and then assigning each letter, digit, and a space to a number. Using a based number system (such as a base three system with pairs of colors representing the digits 0, 1, and 2), each symbol will require a set number of SweetTarts. By representing a space with a SweetTart pattern, all symbols can run together. Thus, a message consists of a stack of SweetTarts that can be rolled in a

piece of paper and transported.

MANY other encryption schemes are possible; however, all schemes must be able to represent the alphabet and in some way communicate numbers. When students design a method, you can ask them the following questions to make sure that they have considered all major aspects.

- Are all letters of the alphabet represented?
- Can they communicate numbers?
- How will they show the separation between different letters, numbers, and words?
- Will they have enough SweetTarts to encode a fifty-character message? Or, what is the average number of required SweetTarts to represent a character?
- Can they transmit (i. e., transport between rooms) their encoded messages without getting the SweetTarts mixed up?

PROVIDING SUCCESSFUL CLOSURE

Each group will have some positive aspect of their system; you need to recognize this. For those groups that faltered, you need to offer constructive comments and praise them for their efforts. A team may not have done as well as they could have because an individual may have forgotten part of their team's code; therefore, you might need to make an extra effort to recognize that team member's positive contributions to their team. You may also need to remind the other members of the team that they are indeed a team and no single person is at fault. Hopefully, you will have enough time for all teams to exercise their processes such that even those that falter will have some success.

Your closure should remind the students that they have simulated a processes for sending and receiving detailed messages. This is the same type of process that satellites use to communicate with scientists and engineers back on earth. It also is the same type of process that computers and other digital devices like CD players, audio and video cassettes, etc. use to store information. The students acted like computer programmers by developing a process to encode and decode messages. They also simulated a computer by remembering their processes. In addition, the students designed a system to transmit their messages, just like a satellite dish and related equipment.

Finally, the students should realize that they have created a new way of communicating, although not directly practical. It may be far fetched, but who knows, someday we might run into a civilization that does communicate with a process using SweetTarts!

SPECIAL SAFETY CONSIDERATIONS

The students will be handling food, which can communicate diseases. We do not advocate students eating the SweetTarts after being touched by so many people.

This project was developed by John Garcelon *and* Dr. David Jenkins.

☐ Return to WAFER ENCRYPTION (CANDY-GRAMS) Home Page.

(3) *Encoding is Not Encryption*, DI Management Services Pty Limited,

pgs. 1-2 (May 13, 2006 = last update).

(http://www.di-mgt.com.au/encode_encrypt.html)

DI Management   DI Management Services is a database design and business systems
consultant based in Sydney, Australia ...more... ▮▱°·   Search | Home

Encoding is Not Encryption

It's unfortunate that the words *encryption* and *encoding* tend to get used as synonyms. In cryptography they mean two different things.

In short:

- Encryption = encipherment = make secret
- Encoding = to convert format, not necessarily securely

Encoding is not encryption. Repeat, encoding is not encryption. Strictly speaking, encryption is an encoding operation, but the term **encoding** is generally used in cryptography to mean that secrecy is not involved.

Encryption

encryption: 1.The process of changing plaintext into ciphertext using a cryptographic algorithm and key. [1]
2. The (reversible) transformation of data by a cryptographic algorithm to produce ciphertext, i.e. to hide the data. [2]

The words **encryption** and **encipherment** mean exactly the same thing.

After encryption, the resulting ciphertext should be indistinguishable from random data and it should be virtually impossible to work out the original plaintext without knowing the key (the exact meaning of "impossible" depending on the cryptographic algorithm used and the length of the key).

We generally need to do some encoding of the data both before and after encryption.

Encoding

encode: 1. To convert data by the use of a code. [3]
2. To format (electronic data) according to a standard format. [4]

Encoding covers many different processes, including:

1. We store textual data encoded in various formats: ASCII, DBCS, EBCDIC, Unicode.
2. We encode ciphertext and other binary data that cannot be printed (that's the stuff with all the funny characters in it) using various formats: base64, hexadecimal, uuencode, binhex.
3. We encode our plaintext in a specific format before we encrypt it. We might convert text stored in Unicode into bytes in a certain order and then add padding.
4. Before using RSA public key encryption or signing, we use an encoding technique on our message (effectively padding plus some control bytes) to make sure the algorithm works properly and to protect against certain known attacks.
5. We store X.509 certificates in DER-encoded format and in PEM format.
6. Compressing the data is also referred to as `encoding', which it is.

Don't get confused

To the uninitiated, the result of an encoding operation may make the data *look* unreadable. Even representing ordinary text in base64 or hexadecimal format can make it harder to read and appear to be stored in a `secret' format. Most people who work in computing can probably recognise hexadecimal encoding of ordinary text. Base64 is much harder. Compressing ordinary text makes it unrecognisable. For example:

```
hex("Hello, world!")=48656C6C6F2C20776F726C6421
base64("Hello, world!")=SGVsbG8sIHdvcmxkIQ==
```

Some text before and after compression using the ZLIB algorithm:

```
000000  68 65 6c 6c 6f 2c 20 68 65 6c 6c 6f 2c 20 68 65   hello, hello, he
000010  6c 6c 6f 2e 20 54 68 69 73 20 69 73 20 61 20 27   llo. This is a '
000020  68 65 6c 6c 6f 20 77 6f 72 6c 64 27 20 6d 65 73   hello world' mes
000030  73 61 67 65 20 66 6f 72 20 74 68 65 20 77 6f 72   sage for the wor
000040  6c 64 2c 20 72 65 70 65 61 74 2c 20 66 6f 72 20   ld, repeat, for
000050  74 68 65 20 77 6f 72 6c 64 2e                     the world.
```

```
000000   78 9c cb 48 cd c9 c9 d7 51 c8 40 a2 f4 14 42 32    x..H....Q.@...B2
000010   32 8b 15 80 28 51 41 1d 2c a2 50 9e 5f 94 93 a2    2...(QA.,.P._...
000020   ae 90 9b 5a 5c 9c 98 9e aa 90 96 5f a4 50 92 91    ...Z\......_.P..
000030   0a 11 d6 51 28 4a 2d 48 4d 2c d1 41 15 d6 03 00    ...Q(J-HM,.A....
000040   86 d1 1f 4e                                        ...N
```

The difference is that there is no security involved. The encoding techniques are not secret. Anyone can carry out the decoding operation. They are not hard to recognize. Be careful.

Further reading

Encryption with International Character Sets
Cross-Platform Encryption
Using Compression with CryptoSys

References

1. NIST Special Publication 800-57 DRAFT (April, 2005)
2. ISO/IEC 10116 (2nd edition): 1997
3. ATIS Telecom Glossary 2000, T1.523-2001
4. The American Heritage® Dictionary of the English Language, Fourth Edition

This document last updated 13 May 2006.

To comment on this **Contact DI Management**.   Return to the **Cryptography page**.   [Top]▲

(X)   <u>**RELATED PROCEEDINGS APPENDIX**</u>

None